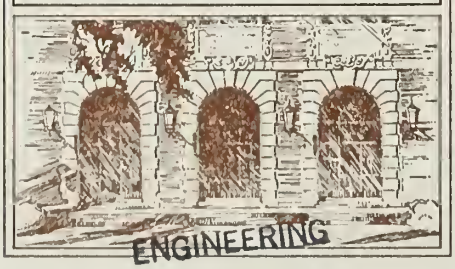
LIBRARY OF THE
UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

510.84
Il63c
no.1-10



AUG 5 1976

The person charging this material is responsible for its return to the library from which it was withdrawn on or before the **Latest Date** stamped below.

Theft, mutilation, and underlining of books are reasons for disciplinary action and may result in dismissal from the University.

UNIVERSITY OF ILLINOIS LIBRARY AT URBANA-CHAMPAIGN

ENGINEERING

CONFERENCE ROOM

JUL 07 REC'D

AUG 04 1983

JUL 15 REC'D

L161—O-1096

Digitized by the Internet Archive
in 2012 with funding from
University of Illinois Urbana-Champaign

http://archive.org/details/tuningofbufferpa00schu

510.84
I63c
no.3

Engin.

CONFERENCE ROOM

ENGINEERING LIBRARY
UNIVERSITY OF ILLINOIS
URBANA, ILLINOIS

Center for Advanced Computation

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
URBANA, ILLINOIS 61801

CAC Document No. 3

THE TUNING OF BUFFER PARAMETERS FOR THE ILLIAC IV DATA MANAGEMENT SYSTEM

by

Stewart A. Schuster

January 15, 1971

CAC Document No. 3

THE TUNING OF BUFFER PARAMETERS FOR THE ILLIAC IV DATA MANAGEMENT SYSTEM

by

Stewart A. Schuster

Center for Advanced Computation
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

January 15, 1971

ENGINEERING LIBRARY

ABSTRACT

The Data Management System provides the lowest level support for the ILLIAC IV Information Management and Analysis System. The design of the Data Management System is based on an experimental technique that is algorithmicly simple but tied significantly to the ILLIAC IV's computational power. Several "tuning" requirements have evolved because of the nature of the system design. This paper presents a Queueing Theory Model of the system. The solution of the model provides the values for the tuning parameters.

TABLE OF CONTENTS

	Page
1. A BRIEF DESCRIPTION OF THE DATA MANAGEMENT SYSTEM	1
2. THE QUEUEING MODEL	4
3. THE SOLUTION	6
LIST OF REFERENCES	10

1. A BRIEF DESCRIPTION OF THE DATA MANAGEMENT SYSTEM

The Data Management System (DMS) is one subsystem of the ILLIAC IV Information Management and Analysis System. The details may be found in reference [3]. A brief description of the DMS follows.

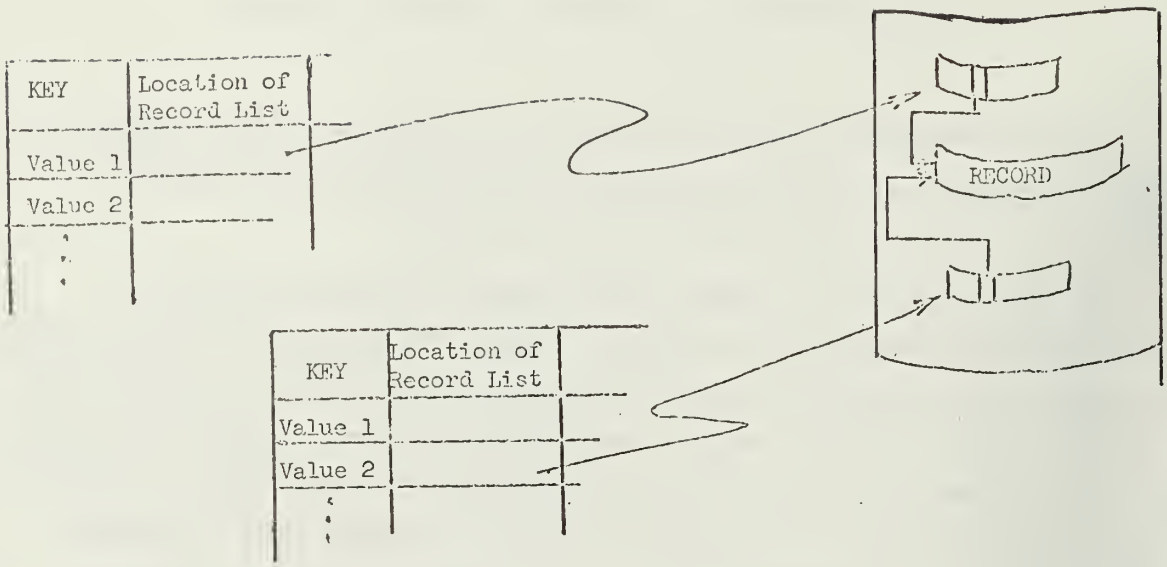
The DMS provides the lowest level support for the Information Management System by performing two tasks. First, it provides an easily usable general purpose file acquisition system to bring files from the archival laser memory to the ILLIAC IV disk. Second, retrieval structures called key elements provide search arguments which are matched according to some relation against other key elements in order to retrieve a subset of records from a file residing on the disk. The retrieved records are examined further by another subsystem. This paper is only concerned with the problems of retrieving records from files on the disk incurred as a result of the design of the DMS.

The designers of the DMS have relied heavily upon the ILLIAC IV processing power and incredibly high I/O transfer rate (about 400 times faster than an IBM 2314 disk pack) to create a simple DMS which has a high access rate but also minimizes queueing of I/O requests.

Other multi-key access systems maintain extensive key tables that must be input, examined, and merged to arrive at a list of addresses where records reside [4]. A series of I/O requests are then made to retrieve these records. Figure 1 is a diagram of this process.

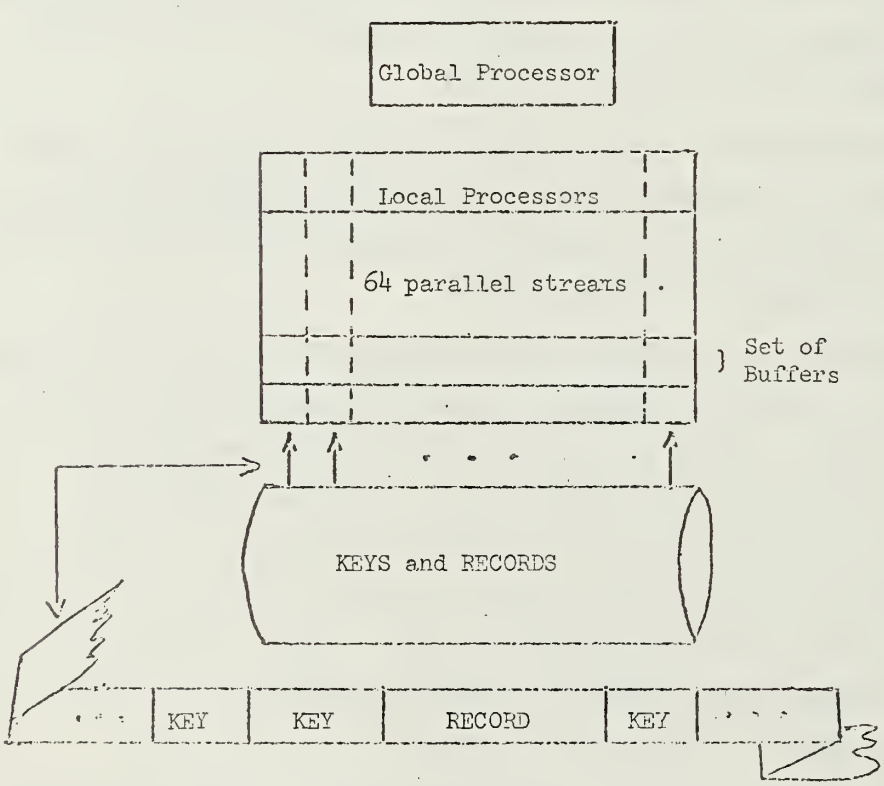
The ILLIAC IV DMS, on the other hand, merges the keys and the records together for any file. Keys that identify a particular record reside on the disk in front of the record. When a record is to be found, the keys are specified and in one I/O request the entire file is flushed through the ILLIAC IV processors by cyclicly filling up buffers in memory. As long as the processor keeps ahead of the transfer rate, all the records can be found for a request in the time it takes to pass the file. Since the I/O rate is so fast, this system responds well ahead of other systems, yet the coding requirements are very small. Figure 2 is a schematic of the process.

Limited core space has been a problem for all the other systems developed for the ILLIAC IV; this system anticipates the same problem.



Conventional Key Table Access to Records

Figure 1



ILLIAC IV Data Stream Access to Records

Figure 2

To keep the DMS simple we have decided not to provide a dynamic storage allocation scheme for incoming buffer loads of records and keys. The problem then is to determine how many fixed-size buffers to maintain in memory and what size they should be. We must provide enough space so that the fixed-sized queue of buffers is rarely full. If the queue becomes full then the I/O must stop, while the processor catches up, and buffers can not be transferred again until the disk makes a complete revolution. If I/O stops too often, the system performance will deteriorate beyond an acceptable level.

Sections 2 and 3 utilize results from Queueing Theory to establish the number and size of input buffers to be maintained in core.

2. THE QUEUEING MODEL

An abstraction of the Data Management System is shown in Figure 3.

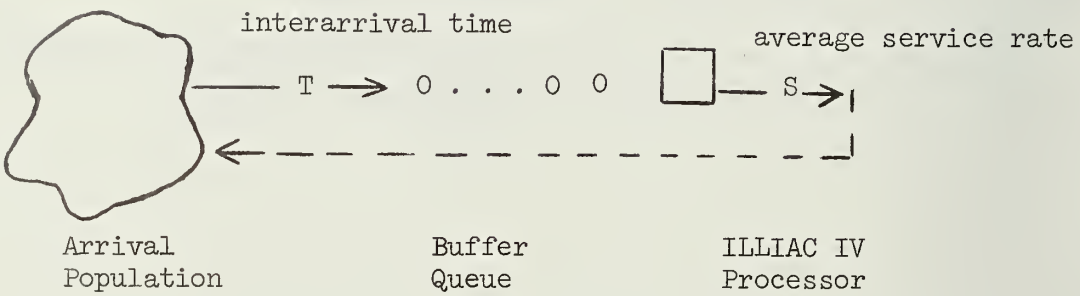


Figure 3

Since the I/O rate is constant, the interarrival time T is constant and it is proportional to the size of the buffer. The larger the buffer, the slower the arrival of a buffer. If we let λ be the arrival rate of one row to ILLIAC IV memory, then $T = k/\lambda$ where k is the number of rows in each buffer. Therefore, the cumulative interarrival time distribution is given by:

$$A(t) = \begin{cases} 0 \text{ for } t < T \\ 1 \text{ for } t \geq T \end{cases}$$

and $T = k/\lambda$.

However, the time required to service or process a buffer depends on the complexity of the buffer and the complexity of the search being performed. The greater the number of keys associated with each record implies more processing time to determine if a particular record is to be retrieved. The greater the number of keys upon which various relations must hold, as determined by the complexity of the search, the longer the processing time required to retrieve records. Since these quantities are unknown prior to a system request, the service times can be described by a probability distribution. Until the system has been built, the service time distribution cannot

be determined. However, until such knowledge can be obtained, we will assume that the service times are distributed according to the exponential density function with mean service rate, S (buffers/unit time). The service rate is also proportional to the size of the buffer. Therefore, until the precise relationship can be determined, we will assume that $S = \mu/k$ where k is the number of rows in each buffer and μ is the average service rate per row. Finally, the service time distribution is given by the following density function:

$$b(t) = Se^{-St}$$

and $$S = \mu/k \quad .$$

The arrival population is finite but very large; all processed buffers are returned to the arrival population so we presume that the input to the processor is infinite. The queue discipline is first-come-first-served.

Reiterating, the problem is to determine, in some optimum sense, the minimum number of buffer spaces needed in core and the size of these buffers. That is, we want to restrict the length of the queue to save core space, but reasonably minimize the possibilities of having to reject an incoming buffer because the queue is full. If this were to happen, the disk would have to make a full rotation before the buffer could be resubmitted. This would cause a serious time delay and performance would deteriorate if this incident occurred regularly.

To determine the size of the restricted queue, we will establish the probability of waiting in the queue longer than time t. We will then determine the probabilities of waiting longer than integral multiples of the average service rate. By choosing an integer, n, which makes the probability small of waiting longer than n expected services we will have good assurance that such a fixed integer number of buffers will not cause I/O to stop.

3. THE SOLUTION

Saaty [1] presents Lindley's derivation and results for the waiting-time (in queue) distribution for a general independent input distribution and general independent service time distribution for a single channel, first-come-first-served queue.

By letting t_n be the arrival interval between the (n)th and (n+1)st unit and s_n denote the service time of the (n)th unit, the waiting time of the (n+1)st unit is established as

$$W_{n+1} = \begin{cases} W_n + U_n, & \text{if } W_n + U_n > 0, \\ 0, & \text{if } W_n + U_n \leq 0, \end{cases}$$

where $U_n = s_n - t_n$. $U(U_n)$ is the cumulative distribution of U_n and the solution given by:

$$U(w) = 1 - \int_0^\infty b(y + w)\, A(y)\, dy$$

where A(y) is the cumulative-interarrival-time distribution and b(y) is the service time density function.

By substituting our functions for A(y) and b(y) it was shown [1] that the probability, $P(\leq t)$, of waiting in the queue less than t is given by:

$$P(\leq t) = 1-(1-p_0)\, e^{-Sp_0 t}$$

and the expected waiting time is

$$Wq = \int_0^\infty t\, d\, P(\leq t) = \frac{1-p_0}{\mu p_0}$$

where p_0, the probability of the system being empty, is the nonzero root of

$$e^{-Sp_0 T} = 1-p_0.$$

Reviewing,

k = number of rows/buffer;

$S = \mu/k$; and

$T = k/\lambda$.

To find P_0 we solve

$$e^{-STp_0} = 1-p_0 \qquad (1)$$

$$e^{\frac{\mu}{\lambda} p_0} = 1-p_0$$

Let the solution to this equation be

$$p_0 = c_1 .$$

The expected waiting time is

$$W_q = \frac{1-p_0}{Sp_0} = \frac{(1-p_0)k}{\mu p_0} = c_2 k, \text{ where } c_2 = \frac{1-c_1}{\mu c_1} .$$

The waiting time is a function of the buffer size, k, and can be minimized by reducing the buffer size to one. However, this would not be optimal for disk storage blocking so this result needs to be examined further within the larger framework of the whole I/O system.

Since 1/S is the average processing time, we establish the solution to the following equation to determine the probability of waiting in the queue longer than n/S, n = 1, 2, ..., .

$$P(< t) = 1 - (1- p_0)e^{-Sp_0 t}$$

$$= 1 - (1 - p_0)e^{-\frac{\mu}{k} p_0 \, n \frac{k}{\mu}}$$

$$= 1 - (1 - p_0)e^{-np_0}$$

$$= 1 - (1 - c_1)e^{-n\, c_1}$$

Note $P(\leq t)$, $t = n/S$ $n = 1, 2, \ldots$, is independent of k, the buffer size.

$$P(> t) = 1 - P(\leq t)$$

$$= (1-c_1)e^{-n\, c_1} . \qquad (2)$$

To determine the probability that an arriving buffer will wait longer than n average processing units, we need to solve equation (2). Therefore, we would choose n buffers as the number of buffers that are needed to give reasonable assurance of not stopping I/O.

To provide a numerical example, a rough estimate of μ (which must be precisely determined later) has been arrived at as

$$\mu \simeq .33 \text{ rows}/\mu\text{sec}.$$

Since the I/O rate is 10^9 bits/sec and there are approximately 4×10^3 bits/row,

$$\lambda \simeq .25 \text{ rows}/\mu\text{sec}.$$

To find c_1 we solve equation (1) giving

$$c_1 = p_0 \simeq 0.45$$

by examination of tables [2]. We solve equation (2) to establish the following table to determine the probability of waiting in the queue longer than n/S, n = 1, 2, ..., .

n	t	P(< t)	P(> t) = 1 - P(< t)
0	0	.45	.55
1	1/S	.65	.35
2	2/S	.78	.22
3	3/S	.86	.14
4	4/S	.91	.09
5	5/S	.94	.06
6	6/S	.97	.03

Therefore, the probability that an arriving buffer will wait longer than 6 average processing time units is only 0.03. Thus, under our assumptions and with our estimates, if we provided 6 buffer spaces in core and used a cyclic buffer technique for incoming buffers, we have reasonable assurance of not having to stop I/O much of the time due to a full queue.

REFERENCES

[1] Saaty, Elements of Queueing Theory, pp. 209-211.

[2] Standard Mathematical Tables, Chemical Rubber Co., 12th Edition.

[3] Schuster, Stewart A. "An Information Management and Analysis System for ILLIAC IV", CAC Document No. 1. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, (December 11, 1970).

[4] Dodd, G. "Elements of Data Management Systems", Computing Surveys, Vol. 1, No. 2, June 1969.

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

ORIGINATING ACTIVITY (Corporate author) Center for Advanced Computation University of Illinois at Urbana-Champaign Urbana, Illinois 61801	2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED 2b. GROUP

REPORT TITLE

The Tuning of Buffer Parameters for the ILLIAC IV Data Management System

DESCRIPTIVE NOTES (Type of report and inclusive dates)

Research Report

AUTHOR(S) (First name, middle initial, last name)

Stewart A. Schuster

REPORT DATE January 15, 1971	7a. TOTAL NO. OF PAGES 15	7b. NO. OF REFS 4
CONTRACT OR GRANT NO. USAF 30-(602)-4144 PROJECT NO. ARPA Order 788	9a. ORIGINATOR'S REPORT NUMBER(S) CAC Document No. 3 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

DISTRIBUTION STATEMENT

Copies may be requested from the address given in (1) above.

SUPPLEMENTARY NOTES None	12. SPONSORING MILITARY ACTIVITY Rome Air Development Center Griffiss Air Force Base Rome, New York 13440

ABSTRACT

The Data Management System provides the lowest level support for the ILLIAC IV Information Management and Analysis System. The design of the Data Management System is based on an experimental technique that is algorithmicly simple but tied significantly to the ILLIAC IV's computational power. Several "tuning" requirements have evolved because of the nature of the system design. This paper presents a Queueing Theory Model of the system. The solution of the model provides the values for the tuning parameters.

DD FORM 1 NOV 68 1473

UNCLASSIFIED

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Searching (General)						
File Maintenance						
Data Structures						
Operations Research						

BOUND BY
HNM

UNIVERSITY OF ILLINOIS-URBANA
510.84IL63C C001
CAC DOCUMENT$URBANA
1-10 1970
3 0112 007263764